BIT-LET

ötlet

SZÁM 1985 JÚNIUS 21

LEHETETLEN-E A LEHETETLEN



A számítógép programozásának, használatának oktatása immáron teljes jogú része a magyar oktatásnak, sőt a magyar könyvkiadásnak, tömegkommunikációnak. A középiskolák után most már az általános iskolák számítógéppel való fölszerelése a téma, úgy az oktatásügy irányításában, mint magukban az iskolákban. A felsőfokú intézményekben folyik tovább a program, amelynek célja, hogy minél több diplomáshoz juttassa el az alapismereteket, s ez a felsőoktatási program már a középiskolai oktatásra, a diákok által a gimnáziumokban, szakközépiskolákban szerzett ismeretekre épít. Azután itt van a tévébasic, amelynek sok gyerekbetegsége volt ugyan, de elvitathatatlan, hogy tömegekhez szólt, s tömegekhez juttatta el a programozás alapismereteit. Itt van a nyár, most táborokban folyik majd tovább az ismeretek terjesztése, a programozás alapjainak elsajátíttatása. A szerkesztőnek mégis hiányérzete van. Hiányérzete van, mert úgy érzi és látja a mindennapi életben, hogy az emberek nagyobbik hányadától még ma is teljesen idegen dolog a számítógép, nem tudják, hogy mire jó, s hogy mennyire egyszerű megtanulni a kapcsolatteremtést a gép és ember között. Az egész oktatási ismeretterjesztési hullámnak van ugyanis egy alapvető fehér foltja. Senki sem törődik azokkal, akik nem akarnak maguknak számítógépet venni, nem akarnak programokat írni, nincs sem idejük, sem kedvük ehhez, matematikai antitalentumnak érzik magukat, s azt gondolván, hogy a számítógép valamiféle matematikai tudást követel használójától, messze elkerülnek minden olyan lehetőséget, ahol ennek ellenkezőjéről győzhetnék meg őket.

Gondoljuk végig, vajon kínál-e oktatásügyünk, tömegkommunikációnk bármiféle lehetőséget azok számára, akik mégis túlteszik magukat ezeken a félelmeken és előítéleteken, s elhatározzák, hogy hajlandók néhány órát rászánni arra az ismerkedésre, amely a számítógép fölhasználó számára szükséges. Nos, számítástechnikával foglalkozó intézmények illetékesei fölszisszenhetnek, mondván, igenis rendszeres tájékoztatókat, tanfolyamokat tartanak olyan gazdasági menedzserek számára, kikben potenciális vevőt látnak. Igaz, ezek az összejövetelek céljukban távol állnak az általam hiányolt lehetőségektől. Ezeknek a tanfolyamoknak, tájékoztatóknak is a szemléletformálás a célja. Ám ezekről a lehetőségekről vajmi kevesen értesülnek, s ráadásul érthető üzleti szempont vezeti a meghirdetőket, amikor elsősorban olyan célközönséget keresnek résztvevőként, akik később vevői lehetnek a rendező cég gépeinek, szoftverjeinek. Egy szó, mint száz a szerkesztő úgy érzi, valami még hiányzik a palettáról. Sem könyv, sem újság, sem film nem szól azokhoz – nem beszélve a gép személyes megismerésének lehetőségéről –, akik némi általános műveltség, vagy átlagon felüli humán jellegű szakműveltség birtokában annyit szeretnének tudni a számítógépről, amennyire nekik szükségük lehet. Emlékszem, néhány hónappal ezelőtt a Magyar Televízióban mintegy a tévébasic alapozásaként hirdették azt az angol tévétől vásárolt sorozatot, amely állítólag ezt lett volna hivatott teljesíteni. Ám a vetítés során kiderült, hogy az első rész első húsz percének alapozása után, sokkal komolyabb közgazdasági és műszaki felkészültséget követelő beszélgetéseket, téziseket sorakoztatott föl a film.

Lehetséges, hogy lehetetlent kíván a szerkesztő, amikor olyan könyvet, olyan tévéműsort reklamál, amely bárki értelmes emberhez szól, amely bárki érdeklődésére számot tarthat, s bárki számára érthetően elmagyarázza a legszükségesebbet a számítógépekről?

Lehetetlen? De gondolta volna-e valaki széles e hazában, hogy a genetika tudományának legizgalmasabb kérdéseit milliók hallgatják-nézik majd esti főműsoridőben? Valószínűleg köznevetség tárgyává vált volna, aki ilyet jósol Czeizel Endre sorozata előtt. Azután mégis megtörtént. Lehet ezek szerint, hogy a lehetetlen mégsem lehetetlen? ***Angyalosi László***

BELÜLRŐL

HÍRODALDAL

szimulált A

Amerikai szakértők számítógéppel szimulálták egy esetleges atomháború következményeit. A gépbe táplált adatok abból indultak ki, hogy az atomrobbanások egymilliárd tonna port juttatnak a légkörbe. A gép kalkulációja szerint a nukleáris robbanófejek egyötöde eltalálná a katonai és ipari célpontokat. Körülbelül tíz napon át hatalmas tüzek égnének. A füst és por következtében a Föld felszínére a napsugárzásnak csupán egy százaléka jutna el. A lehűlés néhány nap alatt több mint 20 fok lenne.

Megtört a jég!

Hosszú vita, húzódozás után megtört a jég az NDK iskola- és oktatásügyében. A következő tanévtől kezdődően a hetedikes diákok számára nemcsak lehetséges, de kötelező is lesz a zsebszámológép használata. A döntést széles körű kísérletek előzték meg, melynek során döntöttek úgy, hogy hat iskolai év után, számolni már biztosan tudó gyerekeknek adják kézbe a számológépet.

Lézernyomtató

A Hewlett-Packard lézernyomtatója nyerte el az elmúlt év végi összegzéseken az 1984 legjobb terméke címet. A LaserJet képességei röviden: 300 pontot rak le egy inchnyi hosszon, ezáltal olyan éles körvonalú betűket tud produkálni, akár egy jó minőségű írógép. Percenként 8 oldalt nyomtat, ez valamivel több, mint 500 sor és mindehhez teljes üzemben 55 dB zaj járul. A készülékhez tetszőleges – nem csak szabványos, perforált papíron használható, sőt akár boríték is címezhető vele. Egyébként a HP LaserJet valószínűleg kijelöli a fejlesztés várható irányvonalát 1985-ben a hasonló profilú vállalatok számára.

Malacgyarapodás

Számítógépes termelésirányítást vezettek be a Bajai Mezőgazdasági Kombinát sertéstelepén. 46 szempont szerint számítógéppel vizsgálják a tenyészállatok genetikai tulajdonságait. Számítógéppel állítják össze az állatok takarmányát és naprakész információt állítanak össze az állatok kondíciójáról és a súlygyarapodásról. Évente mintegy 70 ezer hízott sertést bocsátottak ki. A számítógép alkalmazásától a bajaiak 3–4 százalékos húshozamnövekedést várnak.

BKV adat

Nyolcvan budapesti autóbuszon mikroszámítógépes utasszámláló és forgalmi adatgyűjtő berendezéssel végeznek fontos felmérést. Mérik a járművek sebességét, utazási idejét, a várakozási időket, számolják az utasokat. Rögzítik a piros jelzések előtt eltöltött időt, a forgalmi dugók okozta kieséseket stb. Így megállapítható a szükséges járatsűrítések mértéke, ideje. Később bevezethető lesz a kifüggesztett menetrendszerű járatüzemeltetés.

Kozmetika

A japán Shiseido cég figyelmet keltő szolgáltatást vezetett be egy New York-i áruház kozmetikai osztályán: vállalja a vásárlók bőrének elemzését, azaz egy számítógép speciális berendezés segítségével megvizsgálja a bőr lipid- és víztartalmát, a bőr színét és ennek megfelelő kozmetikai tanáccsal szolgál. Működtetnek egy számítógépes make-up szimulátort is, ami előrejelzi a különböző kozmetikai szerek adott bőrre vonatkozó várható hatását.

ÚJ!



A Dayna Communications nevű cég bejelentette új termékét a MacCharlie-t, amely egy olyan hardver–szoftver készlet, amely lehetővé teszi a Macintosh IBM PC-ként való használatát.
A termék a következőkből áll: egy billentyűzetkiegészítő, két 5,25 inch-es floppy meghajtó egység, memória, egy IBM kompatibilis ROM chip, egy 8088-as processzor. Ezekhez a hardver elemekhez tartozik az a szoftver, amely lehetővé teszi azt, hogy az IBM PC alkalmazások működjenek a Macintosh-on, beleértve az IBM monochrome grafikus lehetőségeket, de a fényceruza alkalmazásának lehetőségét kizárva. Egy floppy meghajtóval és 256 kbyte memóriával a termék ára 1195 dollár, két meghajtóval és 640 kbyte méretű memóriával 1895 dollár.

Táskák

Típus	Képernyő-méret (sorxoszlop)	Súly (kg)	Memória (byte)	Ára (US dollár)
Tandy 100	8x40	1,8	8 K	399
Data General/One	25x80	4,5	128 K	2895
Grid Compass II			512 K	7995
Model 1139	25x128	4,5	+512 K	
Texas Instruments Pro-bite	25x80	4,8	256 K	995
HP Portable	16x80	3,9	272 K	2995
Zenith Pal	16x80	3,5	32 K	–
Sharp PC–5000	8x80	4,4	128 K	1695
Epson Geneva/PX–8	8x80	1,8	64 K	995

A Personal Computing márciusi száma összefoglalót közöl az aktatáska méretű, folyadékkristályos képernyőjű személyi számítógépekről.

Keringő

Színvonalas táncversenyt rendeztek a közelmúltban Szegeden. Az ország különböző részeiből érkezett táncosok az angol keringőtől a rumbáig összesen tízféle táncban mérték össze tudásukat. A rendezvény és a verseny érdekessége volt, hogy az öt pontozóbíró értékelését összegző számlálóbizottság munkáját – Magyarországon először – számítógép segítette.

Primo-kölcsön!

Az MTA SZTAKI, az Elektromodul és a Sárisápi Mgtsz által alapított Microkey Kft. száz darab Primo mikrogépet adott kölcsön fél évre a KISZ Központi Bizottságának. A gépek olyan KISZ-szervezetekhez kerülnek, amelyek vállalják, hogy tanfolyamot szerveznek, klubot alapítanak. A fél év lejártával a KISZ-szervezetek, ha megtetszett nekik a dolog, húsz százalékkal olcsóbban meg is vásárolhatják a Primót.

IBM

Az IBM a növekvő konkurenciaharcban valószínűleg sietni fog új modelljének, a PC II-nek bejelentésével. Várható, hogy az új típus nem lesz nagyobb teljesítményű, mint az IBM PC AT, de az árak és szolgáltatások csábítóbbak lesznek. A központi egység vagy Intel 80186-es, vagy Intel 80188-as processzor lesz.
A PC junior gyártásának leállítása gyártási kapacitást szabadít fel. A cég a személyiszámítógép-vonal felsőbb tagjaira koncentrál, de folytatja a szoftver és periféria gyártását a junior számára.

Computer-M

Ezzel a névvel nyitott a KSH SZÜV ügyfélszolgálati irodát Budapesten a Lenin körúton. Az irodában bemutatják mindazokat a mikroszámítógépeket, amelyekre a SZÜV teljes körű szolgáltatást tud nyújtani. Az irodában megvásárolhatók számítógép-programok, különféle adathordozók és szakkönyvek.

Computer kuckó

A Közalkalmazottak Szakszervezetének Budapesti Bizottsága gazdag programú juniálist rendezett június elsején a Népstadion és Intézményei területén. A színes sport- és kulturális programok mellett számos más szórakoztató érdekesség várta a látogatókat. Volt haditechnikai és hőlégballon-bemutató, motoros sárkányrepülés és így tovább. E speciális programok között kapott helyet a „Komputerkuckó" is, ami nem más, mint a SZÁMALK Szakszervezeti Bizottsága által szervezett mikroszámítógépes játék és vetélkedő műsor. A SZÁMALK (Számítástechnika-Alkalmazási Vállalat) négy nagy sátrat állított fel. Az első, úgynevezett információs sátorban a vállalat szolgáltatásait propagáló kiadványok, szórólapok és szóbeli tájékoztatás mellett a SZÁMALK kiadásában megjelent könyvek voltak beszerezhetők. A második sátorban szórakoztató és vállalatismertető videóprogramokat vetítettek. A harmadik és negyedik sátor szolgált a már említett számítógépes játékok számára. Kicsik és nagyok egyaránt játszhattak a Commodore 64 típusú számítógép játékprogramjaival és összemérhették tudásukat BASIC-programozásban éppúgy, mint a számítástechnikai tesztkérdésekre adandó helyes válaszok tekintetében.
A győztesek értékes díjakat is nyerhettek.

Háziasszonyok

Stockholm egyik élelmiszerboltjában egy nagyszámítógéppel összekötött főzési tanácsadó terminált üzemeltetnek. A „vajon mit főzzek?" kérdésen töprengő vásárló ötletlistát, majd kiválasztás után részletes elkészítési leírást és a hozzávalókra vonatkozó tételes vásárlási listát kap a géptől és a bevásárlást helyben meg is teheti.

Chip kereskedelem

Az Egyesült Államok 1984-ben 1,1 milliárd dollár értékű deficittel zárt a számítástechnikai alkatrész-kereskedelem területén a japán relációban.
Míg az Egyesült Államok 1 milliárd dollár értékű integrált áramkört adott el Japánban, addig Japán 2,1 milliárd dollár értékű alkatrészt adott el az Egyesült Államokban. A MOS memóriák területén az amerikai gyártók közül egyedül a MOSTEK erősítette piaci pozícióját. Az 1984-es rangsorban első a Hitachi és második a NEC.

Szobrász gép

A New York-i Plasztikus Fotográfiai Stúdióban számítógépes szoborkészítő szolgáltatást vezettek be. A megrendelőnek mindössze két másodpercig kell modellt ülnie és ezalatt a szoborfaragó-gép fénysugarakkal letapogatja, majd a háromdimenziós információkat felvevőberendezések kódolt filmszalagra rögzítik. A rögzített és visszaolvasott információk segítségével vezérelt számítógép a „faragó" berendezéssel először egy viaszszobrot készíttet, majd egy körben forgó, húsz késsel működő szerkezet elvégzi a finomításokat az alkotáson. Ezután a kész és élethű szoborról különféle anyagból, tetszés szerinti másolat készíthető a megrendelő kívánsága szerint.

ötlet

GRAFIKUS ZX81

PROGRAM AJÁNLAT

Magyarországon viszonylag olcsó árának köszönhetően az egyik legelterjedtebb számítógép a maga kategóriájában a ZX81. A grafika javítása a HRG-program megjelenéséig szinte csak álom volt. Ez a program azonban csak a PLOT-funkciót tette jobbá, játékok programozásához gyakorlatilag használhatatlan. Az országban már több nagyfelbontású program is található (pl.: FORTY NINER), ahol a programozók megoldották a Spectrumhoz hasonló karakterdefiníciót.
Cikkemben két hónapig tartó munkám eredményét szeretném bemutatni.

Hogyan lehet karaktert csinálni?

1. Hardverbővítéssel
Statikus RAM-okat építünk be a ZX81-be és egy minimális szoftverrel (kb. 20 byte) idetelepítjük a karaktergenerátort. Ez a módszer kényes, házilag nehezen érhető el.
2. Szoftver segítségével
Rákényszerítjük az SCL-chipet (Sinclair Computer Logic) egy több mint 6 K-s képmemória kezelésére. Ez a memória 192 képernyősorból, egyenként 32 „karakter"-ből áll. [Ez 192 (32+1) byte-ot jelent, +1 a sorvégjelző byte.] Az interrupt-regiszter értékének állításával a ROM másmás területeit „nevezhetjük ki" karaktergenerátornak. Alapállapotban I=1EH, ami azt jelenti, hogy a karaktergenerátor az 1E00H címen kezdődik. Hogy ez miért fontos számunkra?
A videómemória bármelyik byte-ját a külső hardver csak indirekt módon, az I-regiszteren keresztül tudja megjeleníteni. A megjelenítés a következő módon történik:
A hardver byte-onként végigolvassa a képmemóriát. Ha olyan byte-ot talál, amelynek 6. bitje egyes, azt sorvégjelző karakternek tekinti. (Ezért nagyon fontos, hogy a képmemória byte-jainak 6. bitje nulla legyen, a sorvégkaraktert, C9H kivéve!) Legyen a videómemória egyik megjeleníthető byte-ja N, az I-regiszter tartalma X. A hardver veszi az X✳256+N✳8 címen lévő byte-ot, N 7. bitjét figyelmen kívül hagyva. Ha N 7. bitje egyes, az előző byte inverzét, azaz 1-es komplemensét veszi. A byte egyes értékű bitjei megjelenítésekor fekete pont kerül a képernyő megfelelő pontjára.
Mivel a ROM-ban egy 512 byte-os lapon minden 8. byte lehet karakter, ezért 64 byte+az inverzeik, azaz 128 byte lehet egyszerre karakter. Mivel 8 bittel $2^8=256$ byte képezhető, ezért teljesen szabadon, bittérképszerűen nem kezelhetjük a képernyőt.
(Megjegyzés: legtökéletesebb módszer a hardver és

```
00 00 C3 9D 40 C3 B7 40
C3 C0 40 C3 C9 40 C3 DB
40 C3 3F 41 84 10 02 00
00 02 06 21 00 67 22 04
40 0E C0 06 20 3A 9C 40
CB B7 77 23 10 F7 36 C9
23 0D 20 EF C9 3E 0C ED
47 DD 21 E1 40 C9 3E 1E
ED 47 DD 21 81 02 C9 06
C0 21 00 67 7E FE C9 28
04 EE 80 77 04 23 10 F4
C9 3A 9A 40 ED 47 C9 21
DF E6 11 21 00 F3 0E FE
06 16 10 FE 06 C0 ED 78
D3 FF 19 CD 09 41 05 C2
F0 40 CD 92 02 CD 20 02
DD 21 E1 40 C3 A4 02 E9
C5 D5 16 00 58 26 00 68
29 29 29 29 29 19 06 00
09 01 00 67 09 D1 C1 C9
F5 C5 E5 47 3A 96 40 2A
10 40 11 06 00 19 16 00
5F 78 B7 28 03 19 10 FD
EB E1 C1 F1 C9 ED 4B 97
40 3A 99 40 CD 0A 41 CD
22 41 3A 96 40 47 3A 9B
40 4F 1A CB 49 20 02 EE
80 CB B7 77 13 D5 11 21
00 19 D1 10 ED C9
```

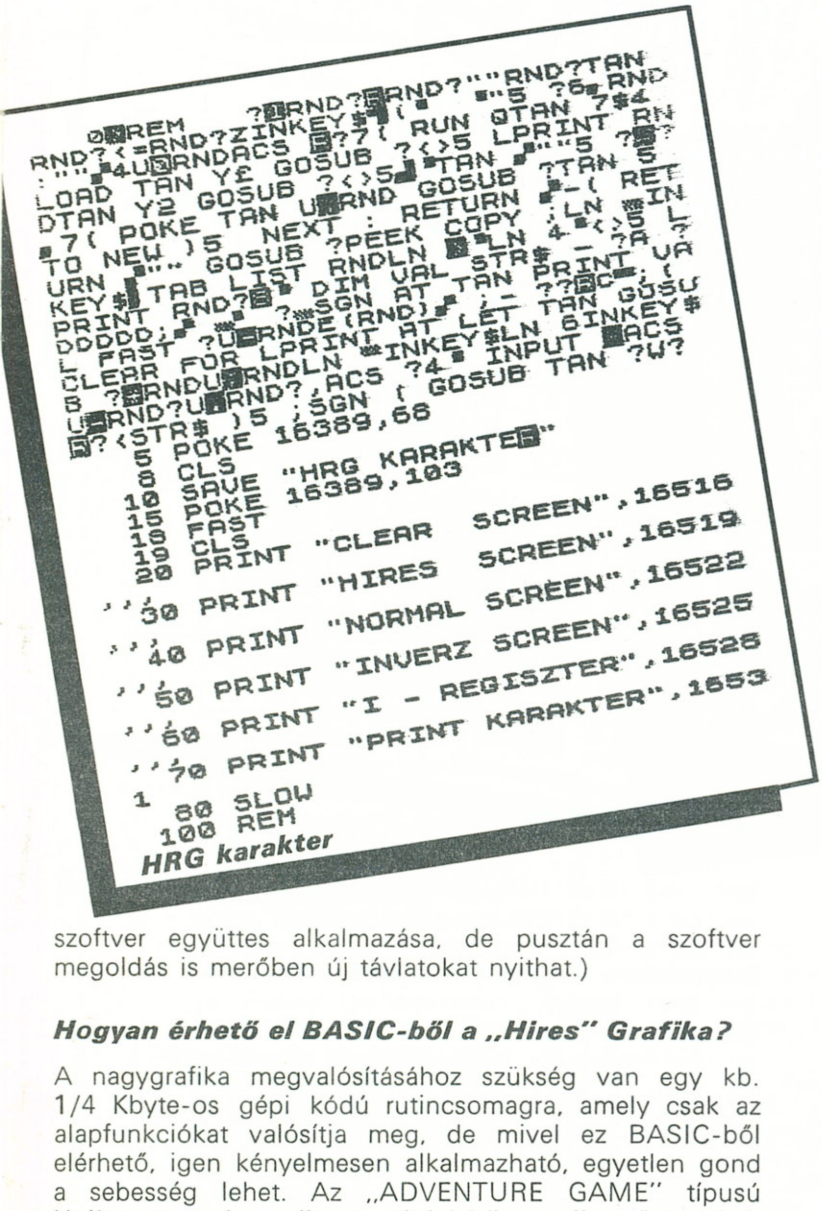
AS.LISTA 1.

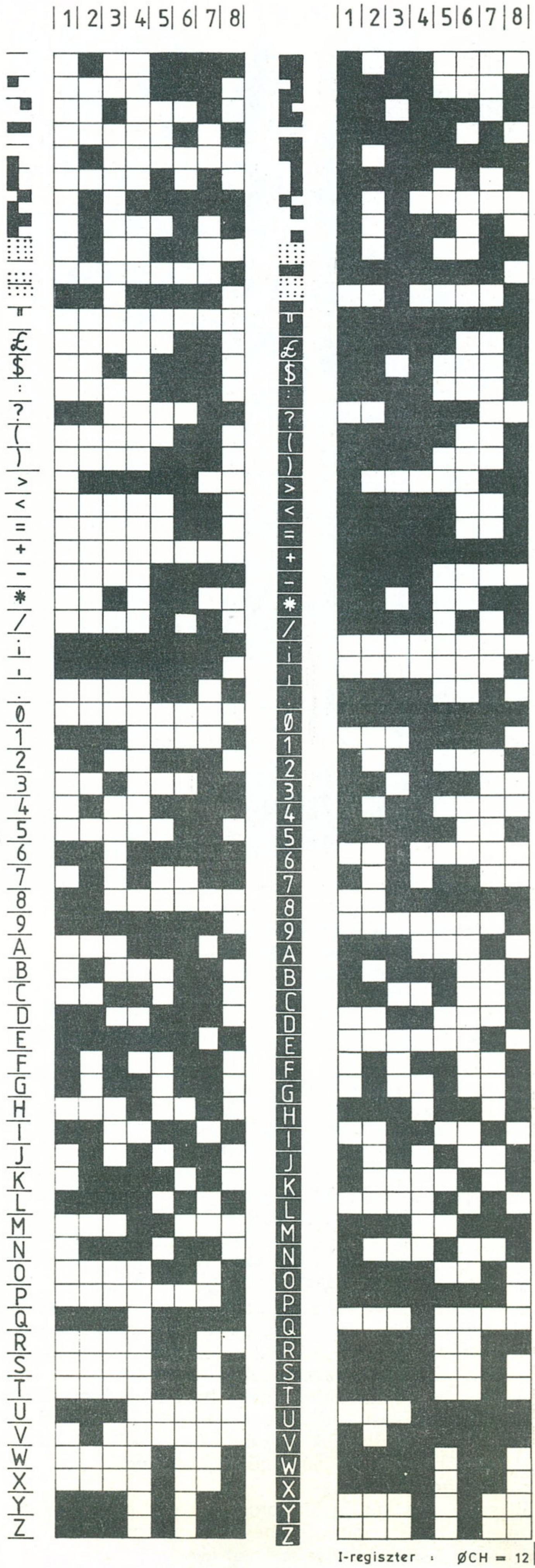
```
00 00 18 18 18 02 88 80
80 00 60 00 00 80 80 88
00 00 80 00 80 80 00 00
88 00 00 00 2A 86 40 7E
67 2F 6F 11 88 40 3E 88
12 13 06 10 AF 29 30 02
3E 60 12 13 78 FE 09 20
04 3E 88 12 13 10 ED 3E
88 12 2A 0C 40 11 88 40
23 06 13 1A CB B7 77 23
13 10 F8 C9
```

AS.LISTA 2.

A szerkesztő azért van,
hogy a lap olyan legyen,
mint amilyenek az olvasói!

Karakter-tábla


```
5 POKE 16389,68
8 CLS
10 SAVE "HRG KARAKTE█"
15 POKE 16389,103
18 FAST
19 CLS
20 PRINT "CLEAR SCREEN",16516
,,30 PRINT "HIRES SCREEN",16519
,,40 PRINT "NORMAL SCREEN",16522
,,50 PRINT "INVERZ SCREEN",16525
,,60 PRINT "I - REGISZTER",16528
,,70 PRINT "PRINT KARAKTER",1653
1 80 SLOW
100 REM
```
HRG karakter



szoftver együttes alkalmazása, de pusztán a szoftver megoldás is merőben új távlatokat nyithat.)

Hogyan érhető el BASIC-ből a „Hires" Grafika?

A nagygrafika megvalósításához szükség van egy kb. 1/4 Kbyte-os gépi kódú rutincsomagra, amely csak az alapfunkciókat valósítja meg, de mivel ez BASIC-ből elérhető, igen kényelmesen alkalmazható, egyetlen gond a sebesség lehet. Az „ADVENTURE GAME" típusú játékprogramokat teljesen gépi kódban célszerű megírni. Mivel a ZX81-tulajdonosok szűk rétege ismeri csak az assembler programozást, ezért először a „HRG–KARAKTER" nevű segédprogramomat ismertetem:

A program egyetlen „REM"-sort tartalmaz, ami magában rejti a kezeléshez szükséges 6 rutint. A program beírása: Az első sorba egy REM utasítást és utána 230 „Z" karaktert írunk. A program többi sorába beírjuk a „zenés ZX81" című cikkben szereplő assembly programot belövő BASIC-rutint (lásd BIT-LET 1983. dec), majd ezt futtatva beírjuk az „AS LISTA 1"-t. Ha ez kész, POKE 16510,0 utasítással kitörölhetetlenné tesszük a „REM"-sort. Kitöröljük a többi BASIC-sort és kimentjük a programot. Kipróbáláskor, vagy már a beírás előtt a POKE 16389,103 és NEW utasításokkal helyet biztosítunk a „nagy képernyő" számára.

I-regiszter : ØCH = 12

ötlet

GRAFIKUS ZX81

PROGRAM AJÁNLAT

```
0 REM    //4 ▒▒ (▒DK Y▒><? RET
   E▒RNDRNDY▒><▒ (▒DK Y▒>E£RND)▒RND
▒   Y▒><( GOSUB Y▒>E£RND) ?W?
URN ▒4▒Y▒><( SAVE TAN ?W?
7▒<,ACS ▒?7<( SAVE TAN
1000 CLS
1010 PRINT "INTERRUPT-REGISZTER
:"
1020 INPUT I
1030 IF I<0 OR I>255 OR I>INT I
THEN GOTO 1020
1035 CLS
1037 PRINT AT 1,0;" 12345678 123
45678"
1040 LET I=2*INT (I/2)
1050 LET ME=256*I
1060 PRINT AT 3,0;"I-REGISZTER :
";I
1065 LET C=0
1070 FOR X=ME TO ME+504 STEP 8
1075 LET Z=PEEK X
1080 PRINT AT 7,0;"TART.:   ";Z;"
";AT 9,0;"KOD  :  (";CHR$ C;")";
AT 9,10;"KOD  :  (";CHR$ (128+C);
")"
1090 POKE 16519,INT (X/256)
1100 POKE 16518,X-256*INT (X/256
)
1110 RAND USR 16516
1120 IF INKEY$="" THEN GOTO 1120
1121 IF INKEY$="1" THEN RUN
1125 LET C=C+1
1130 NEXT X
1140 RUN
2000 CLEAR
2002 POKE 16389,70
2004 CLS
2010 SAVE "KARAKTER-KUTAT▒"
2012 POKE 16389,103
2014 CLS
2020 RUN
```

Karakterkutató

Az általunk beírt program a következő 6 rutint tartalmazza.

1. CLS – A nagy képernyő törlése / RAND USR 16516
A programban mielőtt a nagyképre kapcsolnánk, feltétlenül meg kell hívnunk ezt a rutint. A rutin egyúttal alkalmas a nagykép adott karakterrel való feltöltésére is. A rutin meghívása előtt a 16540 címre kell a kívánt karaktert elvinni, majd a CLS-t hívni. Ezután ne felejtsük el a 16540-es címre az eredeti értékét, 11-et visszaírni!

2. HRG – Nagy képre kapcsolás / RAND USR 16519
A rutint meghívva a 6 kbyte-os képernyő kerül kijelzésre. (Ezt meg kell előznie a CLS-nek!)

3. NOR – Normál képre kapcsolás / RAND USR 16522
A rutin visszaállítja a normál kijelzési üzemmódot.

4. INV – Inverz kép / RAND USR 16525
A nagy kép „színeit" fordítja.

5. REG – I-regiszter állítása / K RAND USR 16528
A rutin a 16538 címen lévő byte-ot az I regiszterbe teszi, gyakorlatilag a karaktergenerátor kezdőcímének beállítására szolgál.

6. PRI – karakter kiírása a nagy képernyőre / RAND USR 16531
5 címen keresztül vezérelhető:
16534 : A karakter nagysága;
16535 : Y koordinátája (0–31);
16536 : X koordinátája (0–191);
16537 : Karakterkód (0–63);
16539 : Inverz-flag (páratlan szám esetén a karakter inverzét adja)

A program használatakor írjunk a 2. sorba egy DIM X$ (N ∗ K) alakú utasítást. N az általunk használt karakterek száma, K a hosszuk. A PRI – rutin az X$ tömb PEEK (16537) SPEEK (16534) elemétől PEEK (16534) byte-ot ír ki a kiválasztott koordinátára, egymás alá.

Térképezzük fel a ROM-ot!

A programozáshoz szükségünk lesz egy alapprogramra (HRG–KARAKTER) és egy karaktergenerátorra (XS tömb). Milyen karakterképek rejlenek a ROM-ban? Ebben segít a „KARAKTERKUTATÓ-PROGRAM". Megadjuk az I regiszter tartalmát, és a program egyenként kiírja az egyes ZX karakterekhez tartozó megjelenítési képet felnagyítva. Így feltérképezhetjük a ROM egész területét, és mindig a legmegfelelőbbet választhatjuk ki.

A program beírása az előzőhöz hasonló módon történik, csak most az első „REM"-sorba 84 db „Z"-t írunk, és az „AS.LISTA 2."-t „lőjük" így be. Ezután beírjuk a karakterkutató többi sorát és kimentjük a programot.

...ÉS a gépi kód?...

Bizony, ez a két BASIC-program csak a logikai, illetve kevés mozgást igénylő játékok programozásához nyújt kellő segítséget. Hosszabb gépi kódú program segítségével már több dolgot sikerült elérnem:

- standard karaktergenerátor
- karakterek mozgatása, villogtatása
- ál-sprite kezelés
- számok kiíratása

Ezek a programok meghaladhatják az 1 K-s, sőt a 2 K-s méreteket is. BASIC-ben nagyon bonyolult dolog beírni őket. Azok viszont, akik rendelkeznek valamilyen assembler-fordítóval sok mindent csinálhatnak. Én már több játékprogramot készítettem így, s ha érdekli a tisztelt programozótársakat az ezekben használt rutinokból szívesen közlök néhányat legközelebb.

Végül néhány jó tanács:

- Betöltés előtt ne hagyjuk ki: POKE 16388,0 ; POKE 16389 , 103 ; NEW!
- A HRG előtt mindig hívjuk meg a CLS-t!
- Ügyeljünk, ha POKE-kal írunk a nagyképre! (kezdőcíme: 6700H=26368; írás x, y koordinátára: POKE 26368+33 ∗ x+y, N)
- Ne felejtkezzünk el az X$ tömb dimenzionálásáról!
- Az I regisztert ne írjuk át a RAM-ba!
- A PRI-rutinnál ne adjunk meg illegális címeket!
- Ha az I regisztert átírjuk (HRG-nél I=0CH), gondoskodjunk a CLS-rutinban a 16540-es címet egy „üres" karakterre átírni!

Kívánok jó kutatást és programozást a ZX81-en.

ifj. Novák István diák, Berze Nagy János Gimnázium



Totódore 64



Újsághirdetések vegyes rovatában találkoztunk először Fekete L. László nevével, aki számítógéppel kidolgozott speciális totókulcsait ajánlotta. Majd később egy riportcikkben olvashattunk róla. Olyannyira felkeltette érdeklődésünket, hogy elhatároztuk, felkeressük és bemutatjuk a BIT-LET olvasóinak.

Fekete L. László tíz éve rendszeres szenvedélyes totózó. Eleinte állami totókulcsokkal játszott. Majd úgy vélte, hogy ezek a kulcsok nem az igaziak, és elhatározta, maga is megpróbálkozik totókulcsok készítésével. Csinált is jó néhány permutációs, vezetéses kulcsot. A kulcsok jók voltak, de az államiakhoz hasonlóan igen drágák. Ekkor egy új ötlete segítségével sikerült olyan kulcsokat szerkeszteni, amellyel 50–60 százalékkal csökkenhetett a befizetési összeg. Az ötlet lényege – amit Fekete maxolásnak nevezett el –, hogy az ugyancsak permutációs kulcsrendszerben maximálta az egymás utáni találatok számát. Ezt megtehette, mivel az egymás utáni sorozatos találatoknak kicsi a valószínűsége. Viszont így jelentősen csökkent a befizetési összeg. Ekkor már másodállásban hivatalos totókulcskészítőként dolgozott a Tanács engedélyével és a Sportfogadási és Lottóigazgatóság tudomásulvételével. Azután egy szép napon kapott ajándékba egy Commodore 64 típusú számítógépet. Mivel sem a programozáshoz, sem az ajándékba kapott géphez nem értett, beiratkozott és elvégezte a Bánki Donát Műszaki Főiskola számítógép-programozói, felsőfokú tanfolyamát. A számítógép megjelenése az életében új fordulatot hozott munkájában is.

A gép alkalmazásba vétele új dimenziókat nyitott meg. Újabb, sok-sok számítást igénylő kulcsötleteit a Commodore, vagy talán „Totodore 64"-gyel már könnyebben meg lehetett valósítani. Az új kulcsok iránt megnövekedett érdeklődés viszont arra késztette, hogy a másodállást a szabadfoglalkozású totókulcs-szerkesztői státusra cserélje fel. A számítógéppel készített kulcsok befizetés/esély aránya tovább javult. Például egy tizenhárom kétesélyes variáció az új kulccsal tízszer kevesebbe kerül, mint a „hivatalossal".

Fekete jelenleg mintegy harminc – Basic nyelven írt – saját fejlesztésű, totókulcskészítő programmal rendelkezik. Most is készül egy új kulcs. Commodore gépe naponta kb. tíz órán át dolgozik legújabb kulcsszisztémáján, melyben tippelni csak annyiban kell majd, hogy a fogadó megsaccolja az 1-esek, 2-esek, X-ek egymáshoz viszonyított arányait. A többi a kulcson és a szerencsén múlik. Ezt az új szisztémát minden eddiginél hatékonyabbnak tartja szerzője.

Megkértük Fekete L. Lászlót, mutassa be egyik kulcskészítő programját olvasóinknak. Ez elől azonban – érthető okokból – elzárkózott. Készített viszont a BIT-LET részére egy (itt látható) úgynevezett tippelő programot, ami szintén jelentősen növelheti a nyerési esélyt. A program által feltett kérdésekre adandó válaszok minden héten megtalálhatók a Sportfogadás című lap aktuális számában. A „magánvélemény"-re adott saját tipp, mintegy húsz százalékban van befolyással a gép által végül kiírt tipp alakulására. Egyébként a 640, 650, 660 sorokban a P-hez hozzáadott érték változtatásával a „magánvélemény" hatása növelhető vagy csökkenthető. Kisebb átalakítással e program futtatható például a ZX81 vagy ZX Spectrum mikrogépeken is.

Fekete L. László elmondta, hogy a közreadott program egy egyszerű megoldás, de számít az olvasók fantáziájára, akik ötleteikkel kiegészítve tovább növelhetik a tippelés esélyességét. E program egyébként egy darab kollektív szelvényen megjátszható tippsort ad. A befizetési összeg mindig más, mivel az adott hétre készített tippek függvényében változik.

Végezetül minden totózó olvasónknak jó tippelést kívánunk és felhívjuk figyelmüket arra, hogyha a programmal kapcsolatban bármilyen problémájuk, megkonzultálandó ötletük van, levélben megkereshetik Fekete L. Lászlót, (1361 Budapest, Pf. 25.), aki szívesen áll rendelkezésükre.

– cs –

Program a következő oldalon ⟶

```
10 PRINT "♥QQQQQ":DIM W$(16)
20 PRINT TAB(9)"*********************"
30 PRINT TAB(9)"*                   *"
40 PRINT TAB(9)"* TIPPELO   PROGRAM *"
50 PRINT TAB(9)"*                   *"
60 PRINT TAB(9)"*********************"
70 PRINT : PRINT : PRINT
80 PRINT TAB(5)"HA MEHET, NYOMJ MEG EGY GOMBOT"
90 GET E$ : IF E$="" THEN 90
100 PRINT"♥"
105 PRINT TAB(4)"HANY MERKOZESRE KIVAN TIPPELNI"
107 PRINT TAB(2)"***********************************"
108 INPUT Q
109 PRINT"♥"
110 FOR Z=1 TO Q
120 P=0
130 PRINT TAB(9)"AZ OTTHON JATSZO CSAPAT:"
140 PRINT TAB(7)"******************************"
150 PRINT TAB(7)"HELYEZESE                 ";:INPUT A
160 PRINT TAB(7)"GOLARANY KULONBSEGE       ";:INPUT B
170 PRINT TAB(7)"EDDIGI GYOZELMEI          ";:INPUT C
180 PRINT TAB(7)"EDDIGI DONTETLENEI        ";:INPUT I
190 PRINT TAB(7)"EDDIGI VERESEGEI          ";:INPUT J
200 PRINT"♥"
210 PRINT TAB(8)"AZ IDEGENBEN JATSZO CSAPAT:"
220 PRINT TAB(6)"********************************"
230 PRINT TAB(6)"HELYEZESE                   ";:INPUT D
240 PRINT TAB(6)"GOLARANY KULONBSEGE         ";:INPUT E
250 PRINT TAB(6)"EDDIGI GYOZELMEI            ";:INPUT F
260 PRINT TAB(6)"EDDIGI DONTETLENEI          ";:INPUT M
270 PRINT TAB(6)"EDDIGI VERESEGEI            ";:INPUT N
280 PRINT"♥"
290 PRINT TAB(6)"A KET CSAPAT EGYMAS ELLEN ELERT:"
300 PRINT TAB(4)"*************************************"
310 PRINT TAB(4)"GYOZELME                          ";:INPUT G
320 PRINT TAB(4)"DONTETLENE                        ";:INPUT K
330 PRINT TAB(4)"VERESEGEI                         ";:INPUT L
340 PRINT"♥"
350 PRINT TAB(13)"MAGANVELEMENY:"
360 PRINT TAB(11)"*****************"
370 PRINT TAB(11)"FIXET KEREK ! ";:INPUT H$
380 IF A>0 AND A<11 THEN P=P+20
390 IF A>10 THEN P=P+10
400 IF B>=0 THEN P=P+20
410 IF B<0  THEN P=P-10
420 IF C>F  THEN P=P+20
430 IF C=F  THEN P=P+10
440 IF C<F  THEN P=P-8
450 IF I>M  THEN P=P+20
460 IF I=M  THEN P=P+5
470 IF I<M  THEN P=P-6
480 IF J>N  THEN P=P-25
490 IF J=N  THEN P=P+1
500 IF J<N  THEN P=P+19
510 IF D>0 AND D<11 THEN P=P-23
520 IF D>10 THEN P=P+23
530 IF E>=0 THEN P=P-16
540 IF E<0  THEN P=P+4
550 IF G>K  THEN P=P+6
560 IF G=K  THEN P=P+2
570 IF G<K  THEN P=P-14
580 IF G>L  THEN P=P+1
590 IF G=L  THEN P=P-10
600 IF G<L  THEN P=P-28
610 IF K>L  THEN P=P+3
620 IF K=L  THEN P=P+1
630 IF K<L  THEN P=P-3
640 IF H$="1"  THEN P=P+38
650 IF H$="X"  THEN P=P+10
660 IF H$="2"  THEN P=P-38
670 IF P<-10 AND P>-80 THEN W$="X2"
680 IF P<-79 THEN W$="2"
690 IF P>15 AND P<80 THEN W$="1X"
700 IF P<-11 AND P<1 THEN W$="1X2"
710 IF P>0 AND P<16 THEN W$="12"
720 IF P>79 THEN W$="1"
725 PRINT "♥"
730 PRINT TAB(3)"A";Z;".TIPP  ";W$:W$(Z)=W$
731 PRINT : PRINT : PRINT : PRINT : PRINT : PRINT : PRINT
733 PRINT TAB(3)"HA MEHET,NYOMJ MEG EGY GOMBOT"
734 GET E$ : IF E$="" THEN 734
735 PRINT "♥"
740 NEXT Z
750 INPUT"KER NYOMTATAST";N$
760 IF N$<>"I" THEN PRINT"VISZLAT!":END
770 OPEN1,4
780 FOR I=1 TO Q
790 PRINT#1,I;". TIPP: ";W$(I)
795 PRINTI;". TIPP: ";W$(I)
800 NEXT I
810 CLOSE 1
820 END
```

POSTA

Két kérdés Polgár Endre (Bp., Columbus u. 52/a) olvasónk leveléből:

„Lassan egy éve már, hogy van egy ZX81-em. Mostanában a gépi kódot és a számítógép működését tanulom és felmerült néhány kérdés, amelyek megválaszolására Önöket kérem.

1. Mi értelmük van az alábbi kódoknak?

– ld a,a ; (d b,b, ld c,c

– and a

– sub a

Számomra értelmetlennek tűnnek, bár az utolsót felfoghatjuk úgy is, mint egy 1 byte-os utasítást, amely 0-t tölt az akkumulátorba.

2. Olvastam, hogy a memóriabővítő letiltja a belső 1 K-s RAM-ot, és vannak olyan bővítők is, amelyek a ROM második 8 K-ját felszabadítják, és szabadon felhasználhatóvá teszik. Az én bővítőm nyugatnémet, és tudomásom szerint ilyet nem csinál, de érdekelne, hogy meg lehet-e oldani az 1 K és/vagy a 8 K használatát utólag?"

Válasz:

1. A sub a utasítás értelmét Ön is megfejtette, igaz erre a célra inkább az xov a utasítást szokták használni. Az and a; ov az utasítások a flag-eket állítják be az akkumulátor tartalmától függően. Akkor alkalmazzuk őket, ha pl. egy ld-vel betöltött adatot szeretnénk megvizsgálni. Pl.:

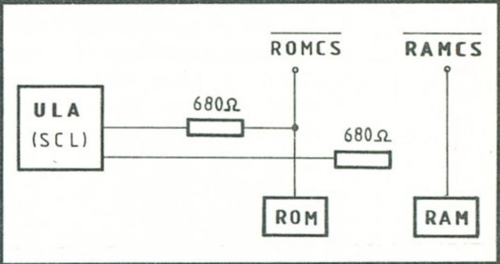
ld a, (hl)

and a

jv z,...

Az ld a,d ; ld b,b stb. utasításoknak tényleg nem sok értelmük van.

2. Az 1 K és 8 K használata megoldható. Erre való a BUSZ, ROMCS és RAMCS vonala, ami a gép belsejében a következőképpen néz ki:

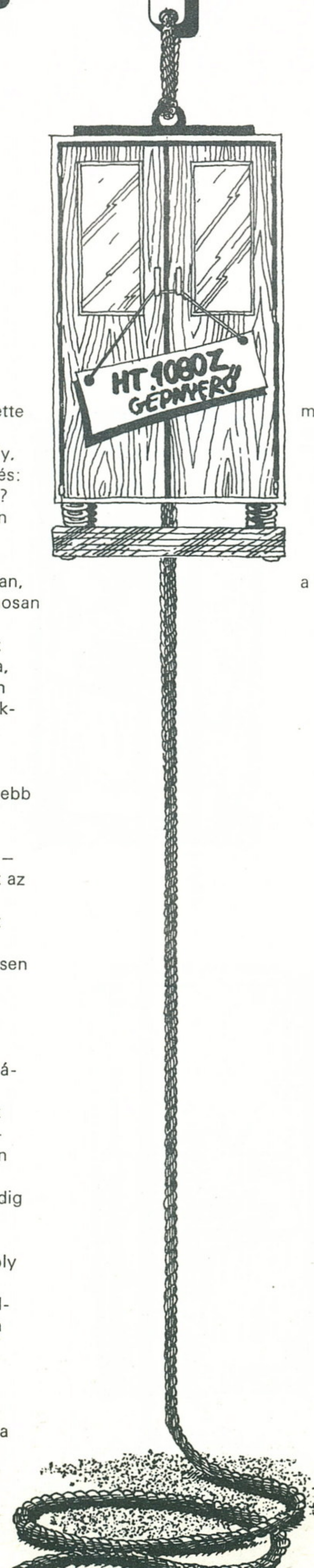


A két ellenállás lehetővé teszi, hogy a ROMCS és RAMCS vonalra egy külső címdekóder jeleit kapcsoljuk, és így a belső ROM és RAM az általunk kiválasztott címtartományba kerülhet. A gyári 16 K-s bővítőkben a RAMCS és a +5V között egy rövidzár van, ez a belső 1 K-s RAM-ot letiltja. A rövidzár megszüntetése után néhány olcsó alkatrész segítségével megoldható, hogy gépünk 17 kbyte-os legyen".

A többi kérdésére levélben válaszolunk.

Csillag Péter

LIFT



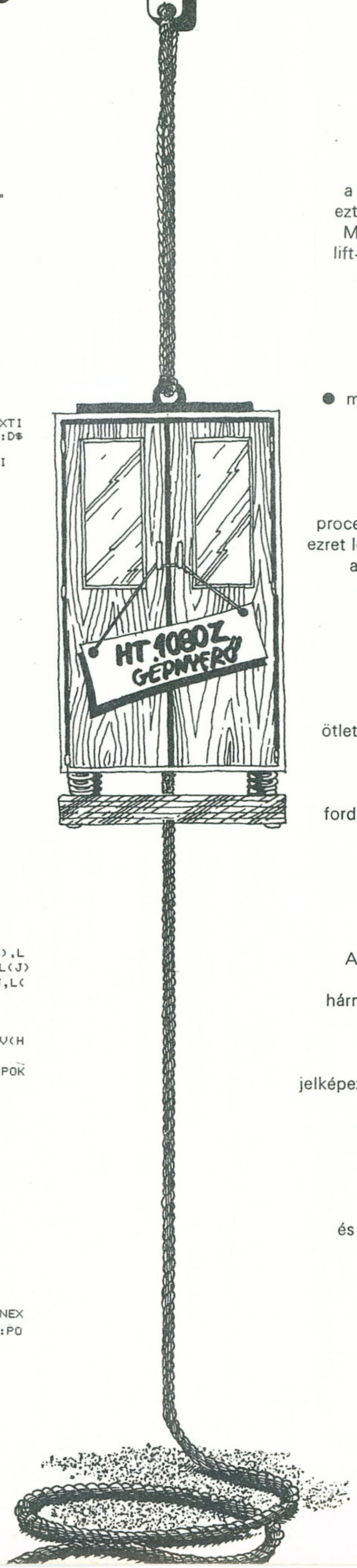
Mint korábban ígértük, most visszatérünk erre az izgalmas feladatra, mely a pályázat során a legnehezebbnek bizonyult. Sok szakkör nem értette meg a feladat lényegét: itt nem csillogó-villogó-csilingelő játékprogramokat vártunk, hanem komoly, használható dolgokat. Többekben felmerült a kérdés: na jó, de mire lehet egy ilyen programot használni? Sőt, lehet-e egyáltalán valamire használni egy ilyen „játék"-nak tűnő valamit? A válasz: igen, lehet. Mióta megjelentek az első számítógépek, persze óriási fejlődés volt az eszközök színvonalában, kapacitásában, gyorsaságában. De ezzel párhuzamosan ugyanilyen fejlődés ment végbe a szoftverkészítés és a számítógépek alkalmazása terén. Hol vagyunk már attól, hogy a számítógépeket csak összeadásra, szorzásra használják? Hála az utóbbi néhány évben egyre növekvő publicitásnak, ma már minden érdeklődő legalább körvonalakban fel tudja mérni, hogy mi mindenre jó egy számítógép. Ezzel szemben úgy érezzük, elég kevés szó esik a szimulációról, a számítógép-felhasználás régóta egyik legjelentősebb területéről (jelentősége a folyamatirányításéval mérhető össze). Pedig ez nemcsak fontos, hanem nagyon izgalmas területe is a számítástechnikának – épp ezért előnyösnek látszik a diákok egy részének az érdeklődését ez irányba terelgetni (mondjuk a játékprogramok felől). Éppen ez volt pályázatunk első és második feladatának a célja.

Mi mindenre használják a szimulációt? Természetesen olyan sok mindenre, hogy felsorolni képtelenség. Még bővebb azoknak a dolgoknak a halmaza, ahol lehetne használni a szimulációt. Hiszen végső soron azt mondhatjuk, hogy minden „komolyabb" új termék gyártását meg kellene előznie egy szimulációs vizsgálatnak. Hogy az micsoda? Nézzük meg néhány példán. Amikor egy fogaskereket terveznek egy adott gépbe, általában a mechanika és a matematika segítségével ki tudják számolni, hogy milyen erők hatnak rá, s meg tudják úgy tervezni, hogy minden ilyen várható hatást kibírjon. Azonban mindig történhetnek váratlan dolgok, beszorulhat valami két fogaskerék közé – s akkor reccs. Bizonyos gépeknél (pl. közlekedés, daruk) ez már igen komoly problémát okozhat. Azonban a világon gyártanak a fogaskerekeknél összehasonlíthatatlanul bonyolultabb dolgokat is, ahol már a várható hatásokat sem lehet mindig kiszámolni. S persze legtöbb helyen fel kell készülni a véletlenekre is, gondoljunk pl. az atomreaktorokra, ahol egy olyan véletlen, amire előre nem számítottak, amit előre nem védtek ki, óriási katasztrófával végződhet! Itt lép be a dologba a szimuláció, amit legegyszerűbben úgy fogalmazhatunk meg, hogy a valóság többé-kevésbé pontos modellezése. A „többé-kevésbé"-t nem lehet pontosan megfogalmazni, de világos egyrészt az, hogy teljesen pontos modellt semmiről sem tudunk készíteni, másrészt az is, hogy a szükséges pontosság annak is függvénye, hogy mit akarunk modellezni, s milyen információkat akarunk a modellből nyerni. Például, ha egy repülőgép szárnyai és teste körül kialakuló légörvényekre vagyunk kíváncsiak, akkor felesleges a motorját az utolsó alkatrészig lemodellezni, elég tudni a motor legfontosabb paramétereit (teljesítmény, max. fordulatszám stb.), másrészt egész pontosan le kell írnunk a gép külső felületét, s jól kell ismernünk a légmozgás törvényszerűségeit. Ha mindezeket „beadtuk" a számítógépnek, kezdődhet a szimuláció. A gép „szimulálhat" szelet, véletlenszerű légnyomást, madarakat stb. – a lényeg, hogy közben „figyelje", hogy mi történik a repülőgéppel, milyen légörvények keletkeznek, s ezeknek milyen hatása van.

A végén a gép „értékeli" a tervezett repülőgéptestet (több szempontból is), s több ilyen értékelés felhasználásával a tervezők tökéletesíthetik, több megoldás közül választhatnak stb. Többen azt mondhatják: hiszen ott van a szélcsatorna! Csakhogy egy szélcsatorna igen költséges dolog, s abba már csak *elkészült* repülőgéptestet tehetnek! Ráadásul vannak olyan dolgok is, melyekhez nincs szélcsatorna, s csak élesben, emberélet(-ek) veszélyeztetésével lehet kipróbálni. Nyilván nem lehet repülőgépet úgy építeni, hogy csinálnak egy nagyjából jó tervet, felépítik a gépet, s ha a berepülő pilóta nem zuhan le vele, akkor gyártanak többet is. (Ugyanis mi van, ha egymás után tíz rossz lesz? Vagy akár egy is?)

Persze az ilyen irányú szimulációs programok rettenetesen bonyolultak. Mi a szimuláció számtalan területe közül egy másikat szemeltünk ki feladatainkhoz. Manapság rengeteg olyan eszközt gyártanak, mely tartalmaz egy vagy több kisebb célprocesszort. Ezeket a processzorokat is gondosan meg kell tervezni, hogy majd az eszközbe beépítve a gyakorlatban jól megállják a helyüket. Ehhez általában nem elég előre jól átgondolni, hogy mit kell tudni az illető processzornak, az „őt" tartalmazó eszköz könnyen kerülhet olyan helyzetbe, amit kihagytak a számításból. Erre nagyon jó példa a lift, ezért is ezt választottuk 2. (nehezebb) feladatunknak. Nem tudjuk, Magyarországon hogyan tervezik meg

LIFT




```
1 CLS
5 PRINT"  KAPCSOLJ 32 KARAKTERES MODBA!"
6 PRINT:PRINT:PRINT"HA MEGTORTENT,NYOMD MEG AZ 'I'-T"
7 IFPEEK(14338)<>2THEN7
10 CLS:PRINT"       LIFT SZIMULACIO":DIMB(3,5)
20 PRINT:PRINT"BILLENTYUZETROL AKAROD IRANYITA-
25 PRINT"NI A LIFTET,VAGY VELETLENSZERUEN"
26 PRINT"JOJJENEK AZ UTASOK(B/V)?"
30 IFPEEK(14340)=64THENP=1:GOTO90
40 IFPEEK(14337)<>4THEN30
50 P=2:FORI=1TO3
60 FORJ=1TO5
70 READK:B(I,J)=K:NEXTJ,I
80 DATA49,50,51,52,53,81,87,69,82,-1,-1,83,68,70,71
90 DIML(4),V(5,3),I(5),G(5),S(5)
100 CLS:PRINT"A LIFT GOMBJAI           UTICELOK":PRINT
110 FORI=5TO1STEP-1:PRINTUSING"#.EMELET";I;
120 IFI<>1THENPRINT:PRINT:PRINT
130 NEXTI
140 FORI=15374TO16334STEP64:POKEI,191:POKEI+7,191:NEXTI
150 A$=STRING$(8,191):B$=CHR$(191):C$=B$+"      "+B$:D$
=B$+B$:B$=D$+"    "+D$:H=1
160 FORI=1TO5:V(I,0)=16469-192*I:G(I)=V(I,0)-11:NEXTI
170 PRINT@846,A$;:PRINT@910,B$;:PRINT@974,A$;
180 RANDOM:FORI=1TO4:L(I)=-16:NEXTI:L(0)=0
190 T$=INKEY$
200 ONPGOSUB890,1160
210 ONPGOTO270,220
220 FORI=1TO5
230 IFI(I)<>0THENR=2
240 NEXTI
250 IFR<>2THEN200
260 R=0
270 IFC<>HTHEN360
280 IF(I(H)AND1)=1THENI(0)=1ELSEI(0)=-1
290 GOTO450
300 FORI=1TO5:IF(I(I)AND1)=1THENC=I
310 NEXTI
320 IFC<>HTHEN360
330 FORI=5TO1STEP-1:IFI(I)=2THENC=I
340 NEXTI
350 IFC=HTHEN200
360 I(0)=SGN(H-C)
370 GOSUB990
380 IF(1.5-I(0)/2ANDI(H))<>0THEN450
390 IFC<>HTHEN370
400 IFI(0)=1THENJ=1ELSEJ=5
410 FORI=HTOJSTEP-I(0):IFI(I)<>0THENC=I
420 NEXTI
430 IFC<>HTHEN370
440 I(0)=-I(0)
450 X=1108-192*H:PRINT@X,"  ";
460 I=1:Y=S(H)
470 IFP=2THEN1430
480 IFL(0)=4THENFORJ=1TO100:NEXTJ:GOTO550
490 IFV(H,I)*I(0)>H*I(0)THEN540
500 L(0)=L(0)+1:S(H)=S(H)-1
510 J=1:C=(C+V(H,I)-I(0)*ABS(C-V(H,I)))/2
520 IFL(J)>0THENJ=J+1:GOTO520
530 L(J)=V(H,I):V(H,I)=0:POKEV(H,0)+I*2,32:POKEV(H,0),L
(J)+48:FORK=1TO100:NEXTK:POKEV(H,0),32:POKEV(H,0)-1,L(J)
+48:FORK=1TO100:NEXTK:POKEV(H,0)-1,32:POKEV(H,0)-6+J,L(
J)+48
540 IFI<YTHENI=I+1:GOTO480
550 IF(Y=S(H))OR(S(H)=0)THEN590
560 IFV(H,1)=0THENA=1:IFV(H,2)=0THENB=3ELSEB=2ELSEIFV(H
,3)=0THEN630ELSEA=2:B=3
570 V(H,A)=V(H,B):V(H,B)=0:POKEV(H,0)+2*A,V(H,A)+48:POK
EV(H,0)+2*B,32
580 IF(V(H,2)=0)AND(V(H,3)<>0)THENA=2:B=3:GOTO570
590 I(H)=0:IFS(H)=0THEN630
600 FORI=1TOS(H)
610 I(H)=(I(H)OR1.5-SGN(H-V(H,I))/2)
620 NEXTI
630 PRINT@X,D$;
640 FORI=1TOL(0)
650 IFL(I)=-16THENI=I+1:GOTO650
660 POKEG(L(I)),42:NEXTI
670 GOSUB990
680 IFPEEK(G(H))=42THEN700
690 IF(1.5-I(0)/2ANDI(H))<>0THEN450ELSE670
700 POKEG(H),32
710 X=1108-192*H:PRINT@X,"  ";
720 FORI=4TO1STEP-1
730 IF(L(I)<>H)OR(L(I)=-16)THEN760
740 POKEV(H,0)-6+I,32:POKEV(H,0)-1,H+48:FORJ=1TO100:NEX
TJ:POKEV(H,0)-1,32:POKEV(H,0),H+48:FORJ=1TO100:NEXTJ:PO
KEV(H,0),32
750 L(I)=-16:L(0)=L(0)-1
760 NEXTI
770 ONPGOTO790,780
780 IFL(0)=0ANDS(H)=0THENPRINT@X,D$;
790 IF(1.5-I(0)/2ANDI(H))<>0THEN460
800 IFC<>HTHEN880
810 IFI(0)=1THENJ=1ELSEJ=5
820 FORI=HTOJSTEP-I(0):IFI(I)<>0THENC=I
830 NEXTI
840 IFC<>HTHEN880
```

a liftek processzorait. De azt tapasztaljuk, hogy ezt jobban is lehetne csinálni. Hogyan? Hát így. Meg kell kérni néhány diákot, hogy írjanak egy lift-szimulációs programot. Természetesen előtte el kell nekik magyarázni, hogy mit kell tudnia a programnak:

- a lift működési algoritmusa legyen külön rutinban, hogy könnyen lecserélhető legyen
- a lift mozgása és a várakozók jól megfigyelhetőek legyenek,
- működjön billentyűzetről is, és véletlenszerűen is, mégpedig úgy, hogy a „véletlenszerűség" paramétereit szabályozni lehessen
- a program maga is végezzen értékelést, pl. számolja az átlagos várakozási időt stb., stb.

Ezek után már nem kell a kigondolt processzort egyből belerakni egy lift-típusba, több ezret legyártani belőle, s ha sok reklamáció érkezett a liftek működésével kapcsolatban, csak akkor terveztetni újat. Nem! A tervezett algoritmust be lehet építeni a programba, kipróbálni, a felmerülő hibákat kijavítani, több irányba kísérletezni, s a legjobban bevált algoritmust építeni bele a liftbe!

Sajnos, a mi kiírásunk nem volt ilyen egyértelműen célirányos, s így, bár sok jó ötlet volt a programokban, mégis sokan szem elől tévesztették pl. a jó megfigyelhetőséget, ami nélkül a program lehet jó játék, de használni esetleg képtelenség. A második fordulót értékelő számunkban leírtuk az értékelési szempontokat. Ezek alapján a kaposvári Munkácsy Mihály Gimnázium szakkörének programját találtuk a legjobbnak (annak ellenére, hogy „kivitelben" akadtak profibb munkák is, a fő szempontoknak ez felelt meg legjobban):

A lift ötemeletes házban működik, 4 személyes és egy emeleten egyszerre csak maximum hárman várhatnak rá. Az ábrázolás fél képernyőre (32 karakteres módra) készült.

I. A véletlenszerűen generált utasok által vezérelt liftnél az utasokat végig egy szám jelképezi, mégpedig annak az emeletnek a sorszáma, ahova menni akarnak.

A földszintet nem ábrázoltuk, hogy ne kelljen a 0-val is utast ábrázolni, ami zavaró lehet.

A liftbe csak azok szállnak be, akiknek az útirányuk a lift irányával megegyezik.

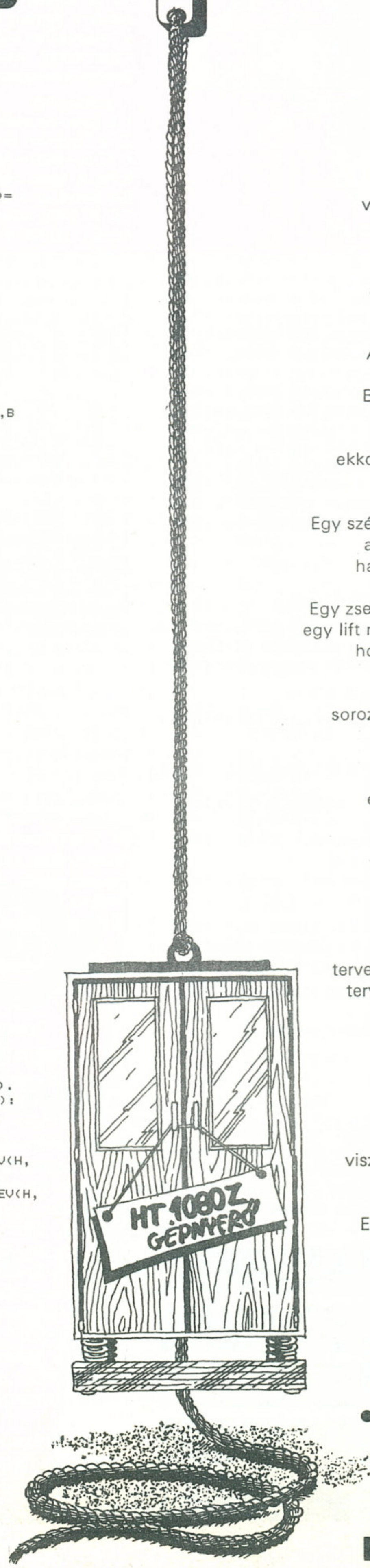
Ha a lift üresen mozog egy hívó felé, és útközben ellenkező irányba készülők vannak, a legtávolabbihoz megy először, így nem kell ezért az emberért később visszamenni, ezáltal időt takarít meg.

A liftben az utasok által megnyomott gombokról a lift mellett balra a csillagok informálnak (azt a gombot nyomták meg, ahol csillag látható).

II. Ha billentyűzetről irányítjuk az embereket (vagyis a liftet), a működési elv lényegében azonos, csak néhány különbség van.

A várakozó embereket „L" vagy „F" betű ábrázolja annak megfelelően, hogy lefelé,

L I F T




```
850 I(0)=-I(0):IF(1.5-I(0)/2ANDI(H))<>0THEN450ELSEI(0)=
0:GOTO300
860 PRINT@X,D$;
870 FORI=1TO50:NEXTI
880 IFL(0)=0THEN370ELSE670
890 K=1
900 M=RND(5):K=K+1
910 IFS(M)=3THENIFK=10THENRETURNELSE900
920 N=RND(5):IFN=MTHEN920
930 S(M)=S(M)+1
940 IFN<MTHENI(M)=I(M)OR1
950 IFN>MTHENI(M)=I(M)OR2
960 V(M,S(M))=N:POKEV(M,0)+2*S(M),N+48
970 IFI(0)=0THENC=M
980 RETURN
990 X=1102-192*H+64*I(0)
1000 FORI=1TO3:PRINT@X-128,C$;:PRINT@X-64,A$;:PRINT@X,B
$;:PRINT@X+64,A$;
1010 IFI(0)=-1THENPRINT@X+128,C$;
1020 FORJ=1TO4:POKE15360+X+J+1,L(J)+48:NEXTJ
1030 ONPGOTO1080,1040
1040 FORK=1TO50
1050 GOSUB1160
1060 NEXTK
1070 GOTO1120
1080 IFRND(15)>5THENR=200:GOTO1100
1090 GOSUB890
1100 FORD=MOTOR
1110 NEXTD
1120 X=X+64*I(0)
1130 NEXTI
1140 H=H-I(0)
1150 RETURN
1160 T$=INKEY$:IFT$=""THENRETURN
1170 D=ASC(T$)
1180 FORM=1TO3
1190 FORN=1TO5
1200 IFD=B(M,N)THEN1230
1210 NEXTN,M
1220 RETURN
1230 ONMGOTO1240,1330,1380
1240 IFL(0)=0THENRETURN
1250 E=VAL(T$)
1260 IFSGN(H-E)<>I(0)THENRETURN
1270 M=1
1280 IFL(M)=6ORL(M)=12THEN1310
1290 IFM<L(0)THENM=M+1:GOTO1280
1300 RETURN
1310 L(M)=E:POKEV(H,0)-6+M,E+48
1320 RETURN
1330 IFS(N)=3THENRETURN
1340 S(N)=S(N)+1:I(N)=I(N)OR2
1350 V(N,S(N))=6:POKEV(N,0)+2*S(N),6
1360 IFI(0)=0THENC=N
1370 RETURN
1380 IFS(N)=3THENRETURN
1390 S(N)=S(N)+1:I(N)=I(N)OR1
1400 V(N,S(N))=12:POKEV(N,0)+2*S(N),12
1410 IFI(0)=0THENC=N
1420 RETURN
1430 IFL(0)=4THENFORJ=1TO100:NEXTJ:GOTO1490
1440 IF(V(H,I)-8)*I(0)<0THEN1490
1450 L(0)=L(0)+1:S(H)=S(H)-1
1460 J=1
1470 IFL(J)<>-16THENJ=J+1:GOTO1470
1480 L(J)=V(H,I):V(H,I)=0:POKEV(H,0)+I*2,32:POKEV(H,0).
L(J):FORK=1TO100:NEXTK:POKEV(H,0),32:POKEV(H,0)-1,L(J):
FORK=1TO100:NEXTK:POKEV(H,0)-1,32:POKEV(H,0)-6+J,L(J)
1490 IFI<YTHENI=I+1:GOTO1430
1500 IFY=S(H)ORS(H)=0THEN1540
1510 IFV(H,1)=0THENA=1:IFV(H,2)=0THENB=3ELSEB=2ELSEIFV(H,
3)=0THEN1540ELSEA=2:B=3
1520 V(H,A)=V(H,B):V(H,B)=0:POKEV(H,0)+A*2,V(H,A):POKEV(H,
0)+2*B,32
1530 IFV(H,2)=0ANDV(H,3)<>0THENA=2:B=3:GOTO1520
1540 I(H)=0:IFS(H)=0THEN1580
1550 FORI=1TOS(H)
1560 IFV(H,I)=6THENI(H)=I(H)OR2ELSEI(H)=I(H)OR1
1570 NEXTI
1580 FORI=1TOL(0)
1590 IFL(I)=6ORL(I)=12THENR=1
1600 NEXTI
1610 IFR=0THEN1690
1620 R=0
1630 GOSUB1160
1640 IFS(H)=0THEN1580
1650 IF(V(H,S(H))-8)*I(0)<0THEN1580
1660 I=S(H)
1670 GOTO1430
1680 GOTO1580
1690 PRINT@X,D$;
1700 FORI=1TOL(0)
1710 IFL(I)=-16THENI=I+1:GOTO1710
1720 C=(C+L(I)-I(0)*ABS(C-L(I)))/2
1730 POKEG(L(I)),42
1740 NEXTI
1750 GOTO670
```

vagy felfelé kívánnak menni (ezt a gombot is nyomják meg a szinteken). A „Le" hívógomboknak a billentyűzeten rendre az „S", „D", „F", „G" gombok felelnek meg a 2., 3., 4., 5. emeleten. A „Fel" gomboknak rendre a „Q", „W", „E", „R" billentyűk felelnek meg az 1., 2., 3., 4. emeleteken.

A lift gombjai a billentyűzeten is ugyanazok: „1", „2", „3", „4", „5".

Beszállás után a lift csak akkor megy tovább, ha a beszállt utasok (balról jobbra) sorban megnyomták, hogy hova kívánnak menni, ekkor az úticéloknak megfelelően számok jelölik az embereket. A továbbiakban a program két része ugyanazt az ábrát mutatja.

Egy szépséghibája van a programnak: ha I. módból a II-be akarunk áttérni, akkor csak úgy lehet, ha leállítjuk a program futását és újra indítjuk.

S most néhány szót az első feladatról.

Egy zsebszámológép megtervezése sokban hasonlít egy lift megtervezéséhez. Nem elég azt megcsinálni, hogy a gép pontosan és jól számoljon, fel kell készülni a felhasználható bármilyen (véletlen vagy akaratos) billentyűnyomás sorozatára, s arra szépen, logikusan kell reagálni. Ennek tervezéséhez szintén jól használható a szimuláció, aki megírta az első feladat programját, rögtön le is tesztelheti „gépét", eldöntheti, venne-e ilyen zsebszámológépet, s ha nem, akkor mit javítana rajta.

Végül felsoroljuk még a szimuláció egy-két területét, remélve, hogy sok diák kap kedvet a számítógépnek eme roppant izgalmas felhasználási módjához.

Egy processzornak nem elég a működését, programját megtervezni, pontosan meg kell tervezni az egész áramkört is. Bizony, áramkörök tervezésénél is használnak szimulációt, sőt erre a célra külön szimulációs nyelvek vannak (pl. LOBSTER).

Szimuláció jól alkalmazható a közlekedés tervezésében is, pl. zöldhullám-tervezésnél. Ezeken kívül jó felhasználási területek még a biológia, kémia s a fizika, ahol a kölcsönhatásokat meghatározó szabályszerűségek viszonylag egyszerűek, a testek eredő mozgása mégis sokszor előre kiszámíthatatlan (l. három-test probléma).

Egyelőre ennyit a szimulációról – de szívesen fogadunk bárkitől ezzel kapcsolatos cikket, véleményt. (S persze azt is, ha valaki leírná a BIT-LET számára, hogy jelenleg hogyan tervezik Magyarországon a lifteket?)

Végül néhány ajánlott könyv:

- Rényi Alfréd: Valószínűségszámítás
- Jávor–Benkő: Diszkrét rendszerek szimulációja
- Nievergelt–Farrar: Matematikai problémák megoldásainak számítógépes módszerei
- SIMULA, CSL, GPSS, SIMSCRIPT, LOBSTER szimulációs nyelvek leírásai.

Király Zoltán



SZAKKÖRÖKNEK!

Lebegőpontos aritmetika a Spectrumon III.

A BIT-LET 16. és 17. számában elkezdtük a Spectrum lebegőpontos aritmetikájának ismertetését. Ezek alapján már elég sok mindent meg lehet oldani, azonban sok dolgot egyszerűsíteni lehet, ha ismerjük a további lehetőségeket. Láttuk, hogy a kalkulátorstackbe milyen módokon lehet betölteni számértékeket. (Csak emlékeztetőül: lehetőség van a BASIC programból, valamint a Z80 regisztereiből ide számokat tölteni.) Felmerülhet azonban az az igény, hogy ha egy hosszabb számítás során egy-egy előre meghatározható (konstans) értéket kellene a stackba betölteni, ne kelljen a kalkulátor üzemből kilépni, a számot a regiszterekből betölteni, majd visszalépni. Természetesen, van erre is lehetőség. A kódsorozatban elhelyezett **34H** kód ugyanis éppen azt jelzi, hogy a következő néhány byte nem vezérlőkód, hanem egy, a stack tetejére töltendő számot határoz meg. A következőkben leírjuk, hogy egy-egy számhoz hogyan lehet meghatározni a leírandó byte-ok értékét: (Mindehhez természetesen tisztában kell lennünk a számábrázolással (l. BIT-LET 16. szám).

1. Előállítjuk a kívánt szám 5 byte-os alakját.

2. Ha a legutolsó byte értéke 00H, akkor azt elhagyjuk.

3. Ha az utolsó byte-ot elhagytuk, és az így maradt utolsó byte is 00H, azt is elhagyhatjuk. Ezt a lépést még egyszer megtehetjük. (Az 5 byte-os alak első két byte-ját egyáltalán nem hagyhatjuk el!)

4. Az 5 byte-os alak első byte-jából kivonunk 50H-t.

5. A **4.** lépésben kapott eredménytől függően két úton haladhatunk tovább:

6A *(Ha a kapott eredmény byte első két bitje 00, és a többi között van 1-es)*
Az előbb kapott értékhez hozzáadunk egy számot a következők szerint:
- 00H-t adunk, ha a 5 byte-ból 3-at elhagyhattunk
- 40H-t adunk, ha az 5 byte-ból 2-t hagytunk el
- 80H-t adunk, ha az 5 byte-ből 1-et hagytunk el
- C0H-t adunk, ha mind az 5 byte megmaradt

7A A **6A** lépésben kapott számot leírjuk a 34H kód után, majd azokat a byte-okat, amelyek a 5 byte-os alak utolsó 4 byte-jából megmaradtak.

6B *(Ha a* **4.** *lépésben kapott eredmény első két bitje nem 00, vagy az összes többi 0.)*
A 34H kód után leírjuk a következő számok valamelyikét:
- 00H, ha az 5 byte-ból 3-at elhagytunk
- 40H, ha az 5 byte-ból 2-t hagytunk el
- 80H, ha az 5 byte-ból 1-et hagytunk el
- C0H, ha mind az 5 byte megmaradt

7B Ezután leírjuk a **4** lépésben kapott eredményt, majd azokat a byte-okat, amelyek az 5 byte-os alak utolsó 4 byte-jából megmaradtak. Látható, hogy a **B** esetben egy külön byte kellett az exponens megadására, míg az **A** esetben ezt egybe tudtuk vonni a mantissza byte-ok darabszámának megadásával. Kis munka után kiderülhet, hogy a **B** esetre a „kis egészeknél" (ahol az utolsó byte mindig elhagyható), valamint a 65535-nél nagyobb és $1.42 \cdot 10^{-14}$-nél kisebb számok esetében kerül sor.
Az itt leírtak talán túl bonyolultan hangzanak, de a következőkben néhány példán keresztül remélhetőleg sikerül megmutatni, hogy ez a valóságban sokkal egyszerűbb. A számok 5 byte-os alakjának meghatározását nem részletezzük, de olvasóinknak javasoljuk, hogy gyakorlásképpen ellenőrizzék azokat!

1. példa: 0.1
1 5 byte-os alak: 7DH 4CH CCH CCH CDH
2, 3 Egyetlen byte-ot sem hagyhatunk el
4 7DH−50H=2DH
5 Az **A** ágon mehetünk tovább
6A Mind az 5 byte megmaradt, így 2DH+C0H=EDH
7A A 34H kód után a következő byte-okat kell megadnunk: EDH 4CH CCH CCH CDH

2. példa: 0.09375
1 5 byte-os alak: 7DH 40H 00H 00H 00H
2, 3 Az utolsó 3 byte elhagyható
4 7DH—50H=2DH
5 Az A ágon mehetünk tovább
6A Mivel 3 byte-ot elhagytunk, a 2DH-hoz 00H-t kell adni
7A A 34H kód után megadandó byte-ok: 2DH 40H

3. példa: 4096 („kis egész")
1 5 byte-os alak: 00H 00H 00H 10H 00H
(figyeljük meg a harmadik és negyedik byte elhelyezkedését!)
2 Az utolsó byte elhagyható
3 A középső byte nem hagyható el, mert az utolsó előtti byte nem 00H
4 00H−50H=B0H
5 Az eredmény első bitje 1-es, így a **B** ágon folytathatjuk.
6B, 7B A 34H után a következő byte-okat kell megadni:
80H B0H 00H 00H 10H
A sorozat további részeiben olyan, újabb kódokról lesz szó, amelyek helyes használatához mindezeket, valamint a Z80 utasításait legalábbis jól érteni kell.

Halász Péter

Ha érted,

akkor már elavult!

Bitton posztulátuma

a naprakészen korszerű

elektronikáról.

Szki Sci-L

proper 16

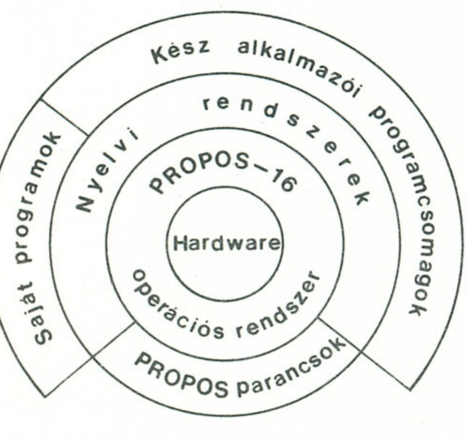
OPERÁCIÓS RENDSZER

A PROGRAM RENDELTETÉSE

A PROPOS–16 a PROPER–16 professzionális személyi számítógépek programkönyvtárának FŐ VONALÁT megalapozó operációs rendszer.
Optimálisan ötvözi a „UNIX-like" és a DOS típusú rendszerek szolgáltatásait.
Egyaránt kielégíti a tipikus – nem számítástechnikus – felhasználók kényelmes kezelés iránti, valamint az igényes szakemberek változatos szolgáltatáskörre vonatkozó igényeit.
A PROPOS–16 fő összetevői:

- PROBIO fizikai szintű erőforrás-kezelés
- PRODOS logikai szintű erőforrás-kezelés
- PROCOM kezelői interface
- PROSYS szervizprogram gyűjtemény.

A szervizprogramok – amelyeknek az alapváltozatban kb. három tucat önálló parancsként használható funkciójuk van – köre „gyári" és saját programokkal is bővíthető.



Fő jellemzők

- A különböző PROPER–16 modellek egységes kiszolgálása.
- Kompatibilitás a DOS-típusú rendszerekkel, valamint a PROPOS–16 korábbi kiadásaival.
- Az aktuális követelmények szerinti konfigurálhatóság, beállítható üzemviteli jellemzők, a hardver konfigurációnak, ill. a szoftver összetételnek megfelelő kiépíthetőség.
- Alap üzemmód (egyidejűleg egyetlen feladat és egyetlen felhasználó kiszolgálása).
- Opcionális üzemmód (a PROMOS opcióval való kiegészítéssel egyidejűleg maximum nyolc feladat, ill. felhasználó kiszolgálása).
- A konkrét igényekhez alkalmazkodó billentyűzet-definiálási lehetőség.
- Flexibilis háttértár kezelés (gyakorlatilag tetszőleges hajlékony-, ill. merevlemezes egységek támogatása; virtuális lemezkezelés).
- Kényelmes és hatékony kezelői interface (menüvezérelt, HELP információval támogatott, a felhasználó által is bővíthető parancs értelmező).
- UNIX-like parancsfeldolgozás: I/O átirányíthatóság, parancsláncolás, egységes paraméterkezelés.
- Hatékony adatvédelmi, adatbiztonsági szolgáltatások: ellenőrzött hozzáférés, mentési, ill. visszaállítási alrendszer, állományok változásának adminisztrálása.
- A nagykapacitású lemeztárak hatékony kezelését biztosító fastruktúrájú könyvtárkezelés.
- Más típusú – pl. CP/M kompatibilis – állományok konvertálásának biztosítása (export, import).

Kezelés

A PROPOS–16 (V.3.0) operációs rendszer egyszerűen kezelhető. A felhasználóknak hierarchikusan szervezett menük formájában kínálja fel a választható parancsokat.
A parancsok használatát, ill. paraméterezését nem ismerők HELP információt kérhetnek, amelyből megismerhetik az egyes parancsok és parancs-csoportok funkcióját, néhány példával illusztrálva.
Az elhagyott paraméterekre a parancsvégrehajtó programok rákérdeznek.
A PROPOS–16 operációs rendszer üzenetei és hibajelzései magyar nyelvűek, de más nyelvű változatok is egyszerűen előállíthatók.

Többszintű kézikönyv-sorozat

A PROPOS–16 (V.3.0) operációs rendszer használatát háromkötetes kézikönyv-sorozat segíti.
Első kötete – *Alapismeretek* – a számítástechnikában járatlan, kezdő felhasználóknak ismerteti a PROPER–16 számítógépek és az operációs rendszer működtetésének alapjait.
A második kötet *Kezelési útmutató*, amely a teljes parancskészlet és a rendszer üzeneteit tartalmazza. Számos példa és összefoglaló táblázatok segítik a megértést.
A harmadik kötet – *Programozóknak* – a PROPOS–16 felépítését és működését írja le. Ebben megtalálható a programból hívható PROPOS rutinok részletes ismertetése is.

Lemezkezelő parancsok

A parancsok e csoportja a lemez egészén fejti ki hatását:

- új hajlékony lemez kezdeti megformálása (FORMAT)
- lemezazonosító címke lekérdezése (VOL)
- hajlékony lemez teljes tartalmának átmásolása (DISKCOPY)
- file-ok mentése egy lemezről egy másik lemezre (BACKUP)
- az elmentett file-ok visszatöltése (RESTORE)
- a lemez paramétereinek kiiratása, „elveszett" területek feltárása (STAT)
- hibás file-ok visszanyerése, javítása (RECOVER)

Könyvtárkezelő parancsok

A fastruktúrába szervezett könyvtárak egyikével dolgoznak az alábbi parancsok:

- egy könyvtár tartalmának kiiratása (DIR)
- a könyvtárak kapcsolatrendszerének megjelenítése (FSYS)
- új alkönyvtár létrehozása (MKDIR vagy MD)
- aktuális könyvtár kijelölése (CHDIR vagy CD)
- alkönyvtár megszüntetése (RMDIR vagy RD)
- keresési utak kijelölése (PATH).

File-kezelő parancsok

Bináris és szövegfile-ok feldolgozásához nyújt segítséget a szervizprogramok e nagy csoportja:

Fejlesztő: SZKI
Számítástechnikai Koordinációs Intézet
1015 Budapest
Donáti u. 35–45.
Telefon: 260-000

- szövegfile tartalmának kiiratása (TYPE)
- kiiratás laponként (MORE)
- szövegfile kinyomtatása (PRINT)
- file-ok másolása és/vagy egyesítése (COPY)
- file nevének megváltoztatása (RENAME)
- file törlése (DEL)
- file-ok összehasonlítása byte-onként (CMP)
- szövegfile-ok összehasonlítása soronként (COMM)
- szövegfile sorainak, szavainak, ill. karaktereinek leszámlálása (COUNT)
- adott karaktersorozat előfordulásainak megkeresése egy szövegfile-ban (FIND)
- szövegfile rendezése (SORT)
- hexadecimális file-kiiratás (HD)
- programfile átalakítása EXE formátumról COM formára (CONVEXE)
- file-attributumok megváltoztatása (CHMOD)
- file-konverzió más operációs rendszerek – PROPOS-8, CP/M – felől, ill. felé (IMP).

Konfiguráló parancsok

A rendszer működését befolyásolhatjuk a következő parancsokkal:

- a képernyő és a nyomtató üzemmódjának beállítása (MODE)
- mágneslemez-műveletek átirányítása másik meghajtóra (ASSIGN)
- standard be-, ill. kimenő adatok átirányítása egy terminálra (CTTY)
- PROPOS–16 rendszerváltozók lekérdezése, ill. beállítása (SET)
- a CONFIG.SYS konfigurációs file lekérdezése, ill. módosítása (CONFIG)
- a menü/help információs file módosítása (MENUSYS).

Batch parancsok

Gyakran ismétlődő parancs-sorozatokat paraméterezhető batch file-okban tárolhatunk, amelyek a következő parancsokkal tehetők hatékonyabbá:

- képernyő üzenetek letiltása vagy engedélyezése (ECHO)
- üzenetek kiírása a képernyőre (REM)
- üzenet kiírása és várakozás egy billentyű lenyomására (PAUSE)
- eltérés a parancsok soros feldolgozásától (GOTO)
- egy parancs feltételtől függő végrehajtása (IF)
- egy parancs ismételt végrehajtása (FOR)
- paraméterek eltolása (SHIFT).

Nyelvi rendszerek

A PROPOS–16 számos magasszintű nyelv használatát támogatja:

BASIC (interpreter és fordító)
PASCAL
C
FORTRAN
COBOL
PL/1
FORTH

A BASIC elsajátítását számítógépes oktatórendszer is segíti.

Felhasználói programcsomagok

A PROPOS–16 operációs rendszerhez kész felhasználói programcsomagok vásárolhatók a következő feladatok megoldására:

- adatgyűjtés, -tárolás és -lekérdezés
- szövegfeldolgozás
- grafikus alkalmazások
- statisztikai számítások

és számos egyéb feladat.

Hardver környezet

A PROPOS–16 minden PROPER–16 modellen, vagy más IBM PC, –XT vagy –AT kompatibilis számítógépen használható. Lehetővé teszi az alábbi konfiguráció kihasználását:

- maximum 1 Mbyte központi tár
- billentyűzet
- fekete-fehér vagy színes grafikus képernyő
- cserélhető (floppy) lemezes meghajtók
- párhuzamos illesztésű nyomtatók
- aszinkron adatátviteli vonalak
- gyors mágneskazettás egység (streamer).

Az operációs rendszer további hardver-berendezések kiszolgálására egyszerűen bővíthető.

Rendelési információ

Termék (változat)	Leírás
PROPOS–16 V.3.0 disztributív rendszer	egy 80 sávos lemez vagy két 40 sávos (2 oldalas) vagy három 40 sávos (1 oldalas)
PROPOS–16 V.3.0 dokumentáció	I. Alapismeretek II. Kezelési útmutató III. Programozóknak

Egy készlet kézikönyv a termék részét képezi, további példányok külön rendelhetők.



Úgy tűnik, szerkesztőségünk eddigi legnagyobb tévedése a VC 20. Hosszas unszolás után álltunk rá ugyanis, hogy egyáltalán foglalkozzunk lapunkban ezzel a szerintünk már divatjamúlt géppel. Azután amikor vallatásunk megjelent, olyan levél- és cikkáradatot indítottunk el, amilyenre még nem volt példa történetünkben. Egyetlen gép – a Primo – az, amely hasonló méretű és mennyiségű hozzászólást, észrevételt váltott, vált ki. Így hát hűen szerkesztőségi jelszavunkhoz (A szerkesztő azért van...), nem szállunk le még egy darabig a VC 20-ról. Ezúttal csak egy kis adalékot és egy előző íráshoz érkezett hozzászólást közlünk, de következő számunktól rövid sorozatban ismertetjük a VC 20 gépi kódú programozásának legfontosabb tudnivalóit.

Tisztelt BIT-LET!

A legutóbbi számban megjelent Tóth Kornél által írt levélhez szeretnék hozzászólni. Kb. fél éve foglalkozom a Z80 programozásával. Egyik barátommal olvasva az újságot, tovább folytattuk Tóth Kornél gondolatmenetét, s úgy érezzük, megalkottuk a számára tökéletesnek tűnő processzort.

1. hogy ne legyen regiszterbőség, ne legyen a processzornak egy regisztere sem
2. teljesen fölösleges olyan sok utasítás, legyen csak egy, mondjuk a NOP, hogy ne legyen nehéz megjegyezni.

Ezekből már következik, hogy teljesen fölösleges az órajel, táp stb., sőt maga a processzor is; úgysem csinál semmit.

Komolyra fordítva a szót: nagyon kétlem, hogy jobb egy 6502-t programozni, mint a vacak Z80-at. Ha így lenne, valószínű, hogy a 6502 terjedt volna jobban el, s nem a Z80.

Néhány aprócska helyesbítés:

1. az IY és IX regiszterpárokat nemcsak 16 bitesként lehet kezelni, holott egyetlen leírásban sem olvastam róla. A DD24 utasítás pl. az IX regiszter felső byte-ját. (Ez biztos.)
2. nem vagyok benne biztos, hogy a 6502 össze tudja fogni a 0. lap 2 byte-ját és mint regiszterpárokat tudja kezelni őket.

Ha így lenne, valószínű, hogy nem láttam volna sok 6502 programban a következő részletet:

```
INC cím 1  3 byte
BNE        2 byte
INC cím 2  3 byte
```

Ha tudna regiszterpárokat, valószínű, hogy nem pazarolnák a programozók így a memóriát.

A Z80 processzorok az általam eddig megismert rendszerekben 2 MHz fölötti órajellel dolgoznak, szemben a 6502 1 MHz-ével, így még a levélben említett összehasonlítás is gyorsabb.

Szerintem a tömbös töltőutasítást sem a memóriakezelés szegénysége miatt találták ki. Valószínűbb, hogy a nagy mennyiségű adathalmaz gyorsabb mozgatása érdekében alkották. (Én legalábbis csak erre használtam.)

Egy Z80

```
LD BC, 700     3 byte
LD DE, 0       3 byte
LD HL, 3CF5    3 byte
LDIR           2 byte
```

programrészletet 11 byte-nál rövidebben 6502-re megírva még nem láttam.

(Ha valaki már megírta, kérem, küldje el nekem.)

A 6502 második indexregiszterének (Y és nem J) átnevezése valószínűleg sajtóhiba. (Sajnos igen – A Szerk.)

Pikács Gábor

(Levelének további részét a Posta rovatban közöljük.)

Örülünk, hogy kedves olvasónk humoránál volt, amikor Tóth Kornél írására reagált. Egyetértünk Önnel annyiban, hogy a VC 20, s még inkább a 6502 processzor védelmében Tóth Kornél kicsit tán elvetette a sulykot. Mi azonban továbbra is úgy gondoljuk, hogy ez minden olvasónknak joga, mint ahogy az erre való reagálás akár vitriolos tollal is. Egy tényre fölhívjuk figyelmét: 65-ös sorozatú processzorokkal több mikroszámítógép készült, mint Z80-assal. Szokásunkhoz híven olvasóink vitájában megőrizzük pártatlanságunkat, s jó magyar szokás szerint nem foglalunk állást. A levelében szereplő tényekkel kapcsolatban azonban megadtuk a lehetőséget Tóth Kornélnak, hogy válaszoljon.

Mivel a 6502 képes MŰVELETEKET végezni a memória byte-jaival, míg a Z80 csak töltögetni tudja (kivéve a BIT-műveleteket), ez az, ami szerintem egyszerűsíti annyira a programozás logikai szervezését, hogy elismerésemet kivívta. A 6502 képes összeadni az Akkumulátor és a memória byte-jának a tartalmát, képes kivonni, összehasonlítani, forgatni, léptetni, továbbá logikai műveleteket képes végezni. Amely dolgokat a Z80 esetében minden esetben a regiszterekbe töltés előzi meg. Én erről az állandó töltögetésről írtam, hogy ezeknél valóban egyszerűbb dolog egy KÖZVETLEN kapcsolat a processzor és a memória között, ha műveleteket kell végezni. A zérólap mint regiszterpár úgy képes működni, mint a Z80 HL regisztere, például a Z80 ADC(HL) utasítása megfelel a 6502 ADC(z)y utasításnak, sőt a 6502 indexként tudja ehhez használni az Y regisztert is. A Z80 esetében a HL-be kell tölteni az akkumulátorhoz adandó címet, a 6502 esetében pedig a zérólapra kell tölteni a kívánt címet. Sőt a 6502 nemcsak összeadáshoz, de további aritmetikai, logikai és hasonlítási műveletekhez is tudja úgy használni a zérólapot, mint a Z80 a HL-regisztert. Természetesen örülök az ilyen ritka szép utasításoknak, mint a LDIR, vagy a BIT-műveletek. De el kell azt is ismerni, hogy a 6502 is tud olyat, amit a Z80 nem tud.

Üdvözlettel: ***Tóth Kornél***

Adalék a VC 20 gépi kódú programozásához

A VC 20-as gép MCS 6502-es mikroprocesszorának gépi kódú programozása megegyezik a 6510-es processzor (C 64) programozásával. A két mikrogép ROM-ja gyakorlatilag azonos, de elhelyezése különböző. A két gépben nem ugyanaz a VIA, illetve VIC chip található, ezért a két gépen a képernyő, szín-, hang-, magnó- stb. kezelése nem azonos.

A BASIC interpreter a C 64-ben $A000–$BFF-ig található, a VC 20-on ugyanez $2000-val magasabban ($C000–$DFFF-ig). A KERNÁL A C 64-ben $E000–$FFF-ig található, ugyanez a VC 20 esetében, annyi különbséggel, hogy az $E000–$E37A közötti címek a VC 20-on $0003-mal alacsonyabban találhatók. Ezen intervallum fölött (E37A-tól) található rutinok szintén azonos típusúak és hatásúak (bár néhány a VC 20 esetében nincs), de a rutinok a két gépben más sorrendben helyezkednek el és így nem azonosak a belépési címek.

A VC 20 esetén a $2000–$7FFF, $0400–0FFF, illetve $A000–$BFFF memóriarészek bővítés számára fenntartottak, a gép alapállapotban ezeket a címeket nem használja. Fontos: 8 Kbyte fölötti bővítés esetén a képernyőmemória átkerül $1000-től kezdődően a színmemória pedig $9400-től kezdődően. A gép karaktergenerátora kontrollregiszterei a $9000–$900F között.

Gépi kódú programokat a BASIC terület végére (vagy elejére) célszerű elhelyezni, a BASIC terület végének (elejének) átállítása után. Tudomásom szerint a VC 20 gépi kódú programozását ismertető könyv magyar nyelven eddig nem került forgalomba, azonban hasznos lehet dr. Úry László: Commodore 64 című könyve és a C 64 Információs kártya. ***Bacsur Kálmán***

ZSÁKBA-MACSKA NYERŐ!



Kérjük levágni és a borítékra felragasztani!
Beküldési határidő: július 15.

BIT-LET-ünk történetében másodszor fordul elő, hogy egy pályázat indításakor még nem tudjuk, hogy mi a verseny tétje, azaz mi a nyeremény. Pályázatunk tehát ismét zsákbamacska névre hallgat. Június–július–augusztus, azaz három hónap három BIT-LET-ében közlünk egy-egy feladatot, s ezek legjobb megfejtői közt sorsoljuk majd ki a nyereményt, amely méltó lesz eddigi pályázataink díjaihoz.

Zsákbamacska nyerőnk ismét egyéni pályázat, ami persze nem jelenti azt, hogy közösségek, kollektívák ne vehetnének részt benne. Ami a feladatokat illeti, az eddigiekhez képest kicsit stílust váltunk. Rovatunk alapkoncepciója, hogy olyan feladatokat tűzünk ki, melyek egyrészt logikai megfontolások, esetleg egy kis próbálgatás után megoldhatók, másrészt van valami közük a számítógépekhez. Eddig főleg játékelméleti, kombinatorikai problémákról volt szó. Most, az alapkoncepciót megtartva, az elemi számelmélet és geometria vizeire evezünk. Ennek szellemében az első feladatunk:

Adott egy 7x7-es négyzet.

1. Bizonyítsuk be, hogy ezt nem tudjuk lefedni 1-nél több darab, különböző (egész) oldalhosszúságú négyzettel! (Természetesen úgy értve a lefedést, hogy a fedő négyzetek nem lóghatnak rá egymásra.)

2. Egy lefedést jobbnak mondunk egy másiknál, ha mindkettő teljes lefedés, s az elsőben kevesebb 1x1-es négyzetet használtunk fel. Adjuk meg a legjobb lefedéseket, s bizonyítsuk be, hogy tényleg nincs náluk jobb! (Itt természetesen már nem feltétel, hogy különböző oldalú négyzetek legyenek, de az igen, hogy oldaluk egész szám legyen.)

POSTA

Bakos Sándorral szeretném közölni (a Posta rovatba írt leveléhez), hogy ha van valamilyen 27XX tipusú EPROM-ja és a programban az ugrások stb. már át vannak irva a megfelelő címre, szívesen beégetem a programot (ingyen). Kérésem csak annyi, hogy a program eredetijét lemásolhassam. Ha kéri, a bővítőt is elkészítem (anyagárban, vagy ha küld anyagot, ingyen). Csupán az esetleges postaköltség megtérítését kérem. (Esetleg a programot is átírom.) Az égetésre vonatkozó ajánlatom mindenkinek szól, aki szeretné EPROM-ban megörökíteni valamely programját. (EPROM-ot nem tudok szerezni!) Eddig elkészített bővítések, ha valakit érdekel:
– QSAVE (SAVE–LOAD gyorsíró) programhoz jelformáló
– 8255 PIO-val felépített interface
– a fent említett EPROM-égető
– 8 bites A/D
(mérés: 50/sec).
Ha valakinek van ZX81 PASCAL fordítója (kétlem), kérem jelentkezzen, nagyon szeretném lemásolni.
Cserébe editor-assembler-monitor; FORTH-fordító; MERGE; TOOLKIT; GRTOOLKIT stb. programot tudok felajánlani, másolási célokra tudom felajánlani a ZXFORTH német nyelvű nyomtatott és géppel-kézzel irott magyar nyelvű leírást.
Maradok a BIT-LET olvasója

Pikács Gábor
Bp. 1085 József krt. 77–79.

Tisztelt szerkesztőség!
Örömmel olvastam a lapukban a bővítéssel foglalkozó eivi törekvést. Amatőrszinten foglalkozom az elektronikával, valamint van egy ZX81-em is. Komolyan foglalkozom a H–RES program EPROM-ba égetésével. A megépítés még kezdeti stádiumban van ugyan, de segítségemet felajánlom a későbbiekben. Készítettem egy modulbox panelt, mely biztosítja a 16KRAM + 2 db egyéb panel csatlakozását az alapgéphez. A sorozat legyártása 2–4 hét múlva esedékes, tehát azután tudnék ilyen panelokat adni, vagy cserélni az érdeklődőkkel. Elkészült még az EPROM-ot tartalmazó panel is, csak még nem lett kipróbálva, az EPROM beégetése még nem megoldott dolog. (Erre is sor kerül kb. 2–3 hét múlva.) Természetesen EPROM-panelt is szívesen küldök az érdeklődőknek.
Sajnos nekem nincsen egyetlen játékprogramom sem, ezért a panelokért majd ilyen programokat szeretnék kérni. Ha valaki előbbre jutott volna az EPROM-os panel építésében, a szerzett tapasztalatokat szívesen átvenném!

Szabó Zoltán
8000 Székesfehérvár, Lövölde út 3. VIII 3.

A levél kézhez vétele és megjelenése óta már el is telt az a 2–3 hét, tehát várjuk az újabb híreket!

Tisztelt szerkesztőség!
Vásároltam egy VZ 200-as Color Computert. Sajnos a dokumentációja elég hiányos. Csak olyan kézikönyvem van hozzá, amely a készülék kezelését ismerteti. Programozói kézikönyv, memóriatérkép sajnos nincs hozzá. Szeretném felvenni a kapcsolatot VZ 200-as tulajdonosokkal. Kérem, akinek a géphez dokumentáció van a birtokában, küldje el fénymásolás céljából.

Lászlófi Ferenc
5440 Kunszentmárton, Kilián út F/20.

Tisztelt szerkesztőség!
Csak néhány hete veszem a lapot, de eléggé megtetszett. Van egy nagy kérésem Önökhöz. Kérem, küldjék el címemre a budapesti mikroklubok listáját. Cimmel együtt. Remélem kérésem teljesitik. Előre is köszönöm

Getse Ferenc
1203 Budapest, Baross u. 3. I. em. 11.

Sajnos, a klubokról pontos nyilvántartást nem vezetünk, de javasoljuk, hogy forduljon a Neumann János Számítógéptudományi Társasághoz (Bp. VI., Báthori u. 16. Tel.: 329-349, vagy 329-390.) Ők készséggel adnak felvilágosítást.

Szerkesztőségünk cime: OTLET Budapest 1986. Szerkesztő: Angyalosi László. Művészeti szerkesztő: Pribelszky Pál
Tördelő: Domokos Imre. Hírrovat: Csányi György. Vallató: Pogány György. Sorvezető: Székely Jenő. Posta: Halász Péter